"Word Pronunciation as a Problem in Case-Based Reasoning"

Wendy G. Lehnert
Department of Computer and Information Science
University of Massachusetts
Amherst, MA 01003
413-545-3639
LEHNERT.UMASS@CSNET-RELAY

ABSTRACT

English word pronunciation is a challenging knowledge acquisition problem in which general rules are subject to frequent exceptions of an arbitrary nature. We have developed a supervised learning system, PRO, which learns about English pronunciation by training with words and their dictionary pronunciations. PRO organizes its knowledge in a case-based memory which preserves fragments of training items but does not remember specific training items in their entirety. After PRO has created a Case Base in response to a training set, it can pronounce novel test words with substantial degrees of success. Test items are processed by generating a search space in the form of a lateral inhibition network and embedding this search space in a larger network that reflects PRO's previous training experience with relevant fragments. Spreading activation and network relaxation are then used to arrive at a preferred pronunciation for the given test item. In this paper we report preliminary test results based on a training corpus of 750 words and a test set of 300 words.

keywords: learning, case-based reasoning, parallel processing
session preference: full paper

Introduction

The rules underlying English word pronunciation are diverse, uncertain, and frequently at odds with one another. For example, when a single vowel is followed by a consonant and a final "e" at the end of a word, a good general rule says to pronounce the vowel as a long vowel. This works for words such as "like," "rope," and "mate." Unfortunately, it doesn't work for words like "love," "move," "give," and "have." But we can't fix our rule by simply excluding the consonant "v" from the general pattern, because many words with a "v" do obey the long-vowel pattern: "save," "gave," "jive," "cove."

Most general rules for English pronunciation are subject to a large number of exceptions which appear to be largely arbitrary. It is tempting to say that one must simply learn each word on a case-by-case basis and forget about identifying a rule-base for this problem. Indeed, once one

has established a "sight-vocabulary," the process of word pronunciation must be heavily aided and influenced by the process of word recognition.

Yet there is evidence that children learning to read rely heavily on a facility for phonological recoding which allows them to move from the written word to a pronunciation of that word before achieving recognition of the word [Bradley & Bryant 1983, Doctor & Coltheart 1980]. Other researchers have argued that phonological recoding is a necessary component of skilled reading as well [Gough 1972].

We have implemented a model of English word pronunciation in the form of a computer program called PRO. PRO learns to produce phonological encodings (pronunciations) for isolated input words by first training on word/pronunciation pairs. All memory structures utilized by PRO are created automatically during training, and PRO implements a model of phonological encoding that is fully independent of the word recognition problem.[1]

PRO operates by creating three types of memory structures in response to its training sessions: (1) the Hypothesis Base, (2) the Case Base, and (3) the Statistical Base. These three levels of memory are hierarchical insofar as the Case Base draws its components from the Hypothesis Base, and the Statistical Base is predicated on the existence of both the Hypothesis Base and the Case Base. As training goes on and memory expands at each level, we do see some memory interactions that shape subsequent memory expansion. For example, information from the Statistical Base is capable of influencing expansion at the level of the Hypothesis Base. But for the most part, memory grows with simple dependencies: the bigger the Hypothesis Base, the bigger the Case Base, and the bigger the Case Base, the bigger the Statistical Base.

Building the Hypothesis Base

The Hypothesis Base consists of simple associations between graphemes [Coltheart 1978] from the input word and phonemes in the target representation. A specific hypothesis in memory tells us that a particular string of letters has resulted in a particular phoneme at least once during training. The mapping defined by the Hypothesis Base from the set of set of all substrings of letters to the set of all phonemes is neither totally defined nor well-defined. Not all substrings need map to a phoneme, and when one does, it may map to any number of phonemes. For example, consonants and consonant combinations are generally less ambiguous than vowels and vowel combinations, so we will see more hypotheses associated with vowels than consonants.

To illustrate the idea of a hypothesis, consider the following segmentation of the word "action" along with its corresponding pronunciation:

A	C	TI	O	N
ă	k	sh	∂	n

When this segmentation is mapped against the target pronunciation, we can identify five underlying hypotheses (associations between graphemes and phonemes): A/a, C/k, TI/sh, O/∂,

[1] Another system that is very similar to PRO in its broad design is MBRtalk [Stanfill & Waltz 1986], although MBRtalk uses a very different form of memory access. MBRtalk runs on a Connection Machine whereas PRO can run reasonably in a Common Lisp environment with 4M of memory.

and N/n.

However, if this were the first word PRO encountered in a fresh training session, PRO would have no way of knowing that this is the correct segmentation of the input string. There are five ways of partitioning this six-letter word into five substrings, and PRO would be unable to decide which is best. In general, PRO consults its existing Hypothesis Base in order to determine how an input word should be segmented and matched against its target pronunciation. After generating all possible segmentations of the input word which map onto the target pronunciation, PRO checks its known hypotheses to see if any one of these segmentations results in more known hypotheses than any other segmentation. If there is a unique winner, that is the segmentation PRO picks. In the event that there is no unique winner, PRO narrows its candidates to whichever ones did maximize known hypotheses. Of these remaining segmentations, PRO checks to see if any of the new hypotheses being proposed are "close fits" against known hypotheses. A hypothesis *string1/pheneme1* is a close fit against *string2/pheneme2* if (1) *pheneme1* = *pheneme2*, and (2) *string2* is a substring of *string1*. If one of the remaining segmentations supports more "close fits" than any other segmentation, then PRO identifies that one as the preferred segmentation. In most cases, PRO will be able to identify a preferred segmentation on the basis of these two filtering mechanisms. But in the event that there is still more than one segmentation which maximizes both known hypotheses and close fits to known hypotheses, then PRO consults its Statistical Base and determines which of the remaining segmentations contains hypotheses that are used with greater frequency. In this manner PRO can identify a preferred segmentation to map against the targeted phonemes.

Once a preferred segmentation is in hand, PRO consults its existing Hypothesis Base to see if all the hypotheses in the segmentation are known hypotheses. If any hypothesis is new, we index it under its grapheme and the Hypothesis Base acquires a new hypothesis. Having updated the Hypothesis Base, PRO then goes on to update the Case Base and the Statistical Base.

Building the Case Base and the Statistical Base

Given a segmentation for a training item, we can take a series of "snapshots" of the resulting hypothesis sequence where each snapshot contains exactly three consecutive hypotheses. The longer the segmentation, the more snapshots we will need to cover the full sequence. Each snapshot of three hypotheses then constitutes a case for PRO's Case Base. Cases are indexed under both the leading hypothesis and the trailing hypothesis, and all cases with a common index are organized in a tree structure. The Case Base therefore contains as many trees as there are hypotheses in the Hypothesis Base, and each tree may contain a few or a large number of cases depending on how much of PRO's training has been involved with the indexing hypothesis. All the trees have a depth of three hypothesis nodes and each complete branch in a tree corresponds to a single case in memory.

When a candidate case is taken from a "snapshot" of a training item, we check to see if that case has been recorded in the Case Base before. If it has, we do not need to alter the Case Base. If it hasn't, we grow a new branch in the appropriate case trees (one for the leading index and one for the trailing index). This dual encoding of each case allows us to maintain a more informative Statistical Base than would otherwise be possible.

The Statistical Base shadows the Case Base by maintaining frequency data for all the nodes in all the case trees. When a case is encountered during training, we add it to the Case Base if needed, and update each of the six tree nodes associated with that case in memory. For example, the case (A/a C/k TI/sh) will update the root node associated with A/a, the node that says how many times A/a has been followed by C/k, and the node that records the frequency of A/a followed by C/k followed by TI/sh. Going backwards, we update the node that records instances of TI/sh, then the node recording TI/sh preceded by C/k, and the node recording TI/sh preceded by C/k preceded by A/a. Note that the two terminal nodes should agree on the number of times a given sequence has been seen before, but the root nodes and intermediate nodes will generally maintain different frequency counts depending on whether the sequence is being traversed forwards or backwards.

Test Mode: Finding a Word Pronunciation

When PRO receives an input word in test mode, it begins by creating a search space of possible pronunciations based on all available hypotheses in memory. Each path in this space beginning with the "start" node and ending with the "end" node corresponds to a possible pronunciation for the word (see Figure 1). For example, if the search space shown in Figure 1 is generated in

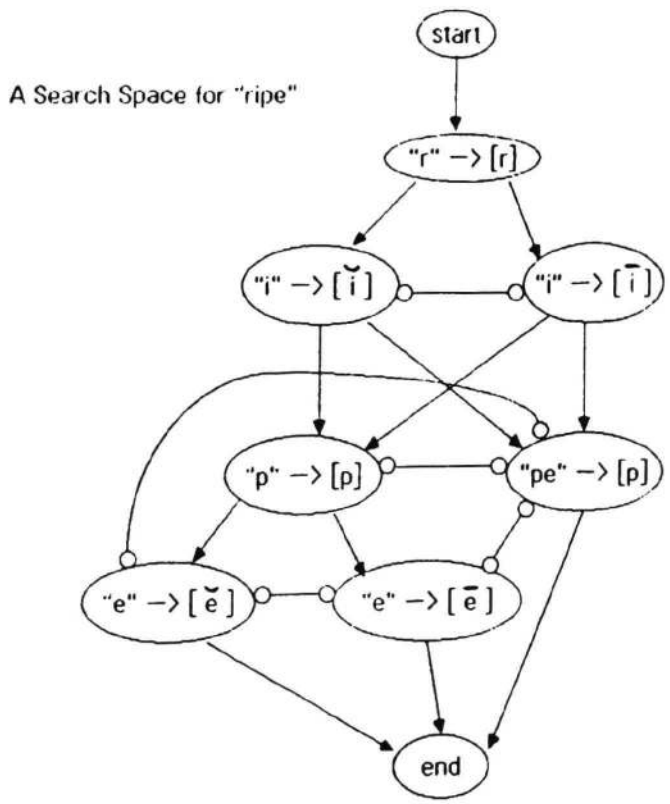

Figure 1: A Search Space using Available Hypotheses

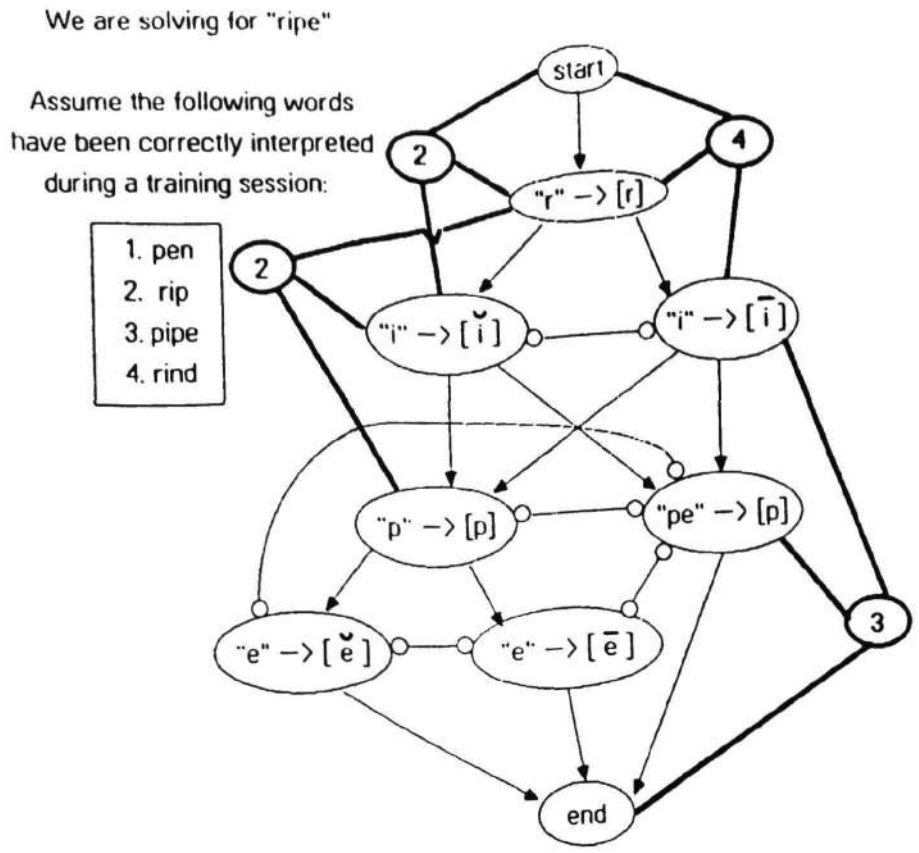

Figure 2: Adding Context Nodes from the Case Base

response to the word "ripe," we have six possible pronunciations for "ripe" corresponding with the six possible paths through that space. There are two possibilities for "i" (long vs. short) and three possibilities for "e" (long vs. short vs. silent). This search space is further structured as a lateral inhibition network where instances of hypotheses which share common characters from the input word inhibit one another. In figure 1 we see that the silent e is represented by the hypothesis PE/p. Since this hypothesis contains two input characters, it is in competition with P/p as well as E/e and E/e. We therefore find three inhibition links connecting PE/p to its three competitors.

Thus far we have described a search space derived from the Hypothesis Base alone. To bring in the effects of the Case Base, we must now add context nodes representing relevant cases (see Figure 2). Each context node corresponds to a case which matches a subpath within the search space. In Figure 2 we see how four context nodes can be added to the graph on the basis of three previous training items ("rip," "pipe," and "rind.") The fourth training item ("pen") is not associated with any cases relevant for this search space. By attaching the resulting context nodes to the search space where each case applies, we see how PRO's previous training with

"rip" reinforces a short "i", while exposures to "pipe" and "rind" reinforce a long "i". "pipe" also reinforces a silent "e" but no context nodes reinforce a short "e" or a long "e" at the end of the word.

The context nodes provide positive activation for a spreading activation algorithm, and the inhibition links counter this activation as it propagates through the network. Context nodes are initialized with a value based on frequency data in the Statistical Base, and a relaxation algorithm is then applied throughout the search space [Feldman & Ballard, 1982]. These networks tend to stabilize within 30 iterations, and a preferred path through the network is sought by maximizing MIN [A(N1),...,A(Nk)] where A(Ni) is the activation level associated with the ith node in a given path. In general, a unique path of maximal activation can be found and this is then the pronunciation PRO returns. If there is more than one path of maximal activation, PRO selects one of the maximal paths randomly and returns that pronunciation.

Experimental Results

We have run PRO on a training vocabulary of 750 words and tested it on a total of 300 words. 200 of the test items appear in the training set and the remaining 100 test items are "novel" words not encountered during training. We have also collected data from a modified version of PRO which substitutes a random search algorithm in place of PRO's spreading activation/relaxation algorithm. The modified version builds the same search space of possible pronunciations, but picks one randomly instead of structuring the network with additional context nodes and relaxing the net to find a pronunciation with maximal activation. The results of this random algorithm provide us with a baseline that reflects the power of the Hypothesis Base operating in the absence of the Case Base and the Statistical Base.

Test trials were run at three distinct times during training: after PRO had trained on 250 words, after 450 words, and after all 750 words. The training corpus contained words ranging from three to eight letters in length, and the test sets contained words ranging from three to five letters in length. We collected separate data for "familiar" words (the first 200 items in the training set), and "novel" words. The "familiar" items included 50 three-letter words, 100 four-letter words, and 50 five-letter words. The "novel" items included 50 four-letter words and 50 five-letter words. The modified version of PRO was run on only "novel" test items, but at the same points during training as the regular PRO runs.

Table 1 shows the results of these nine test runs. The x-axis represents the amount of training PRO has had in terms of the number of target phonemes processed. After 750 words, PRO has examined 2731 phonemes. The y-axis shows the percentage of phonemes PRO correctly identifies during test runs. The top curve shows PRO's performance on "familiar" words, the second curve shows PRO's performance on "novel" words, and the bottom curve shows the baseline performance of the random algorithm on the same set of "novel" words that we tested with PRO in tact.

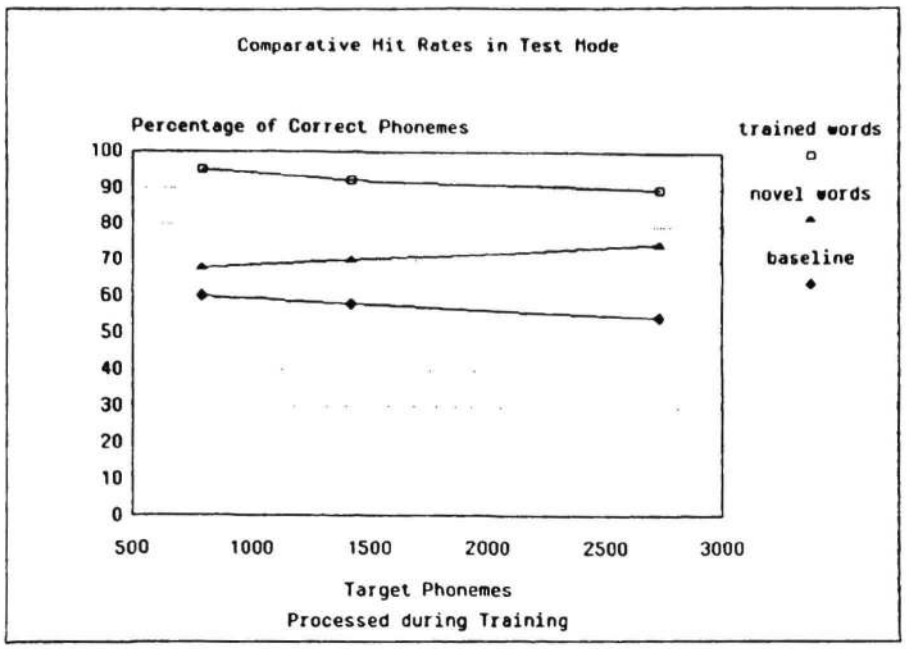

Table 1: Three Test Runs during Training

We can see that PRO is better with words it has seen during training compared to novel test words, although the distance between these two curves does diminish as PRO trains more. It makes sense for PRO's performance on familiar words to degrade somewhat because PRO's Hypothesis Base and Case Base are expanding as PRO trains. As more conflict arises between a larger number of competing cases, we might expect to see the increased error rate found in the top curve. These factors explain why PRO's performance drops off a bit (from 95% to 90%) for familiar words. This same factor must influence PRO's performance on novel words as well, but here we see a gradual increase in performance (from 66% to 75%). For novel words, the increasing search space complexity is overridden by the fact that PRO takes advantage of its growing Case Base effectively.

If we assume that these two curves eventually level out, we can estimate PRO's eventual level of competence across all test items as being somewhere in between the two asymptotes. It is safe to assume that these two curves won't cross, so they must level out somewhere between 75% and 90%. Extrapolating from our existing data, we would estimate that PRO will eventually average around 86% on all test items (this is where the two curves would intersect if they continued at their present rate of change).

To see how much of PRO's performance can be attributed to the Case Base and Statistical Base, consider the baseline curve provided by the random version of PRO. Here the hit rates drop from 60% to 55%. This is what PRO can do if it only accesses knowledge from the Hypothesis Base in order to construct a search space of possible pronunciations. Once this search space

has been constructed, we can pick a pronunciation from the space at random. According to our randomized simulation, any given phoneme within that pronunciation then has a 55% chance of being correct after PRO has trained on 750 words. Once again, it makes sense for this curve to diminish since the Hypothesis Base is growing over time. More hypotheses in memory will result in more possible pronunciations for each test word. If we assume that this baseline levels out at the same time our two performance curves level out, then we can estimate that the baseline will level out around 50%.

From our best estimates, we then can then conclude that PRO's Case Base and Statistical Base will result in a significant improvement over the Hypothesis Base alone (86% vs. 50% hit rates on each phoneme). Moreover, these levels of performance should stabilize once PRO has reached a saturation point in its training.

PRO's Memory: Facts and Conjectures

We have gathered additional data in conjunction with our training session to see what PRO's memory looks like after 750 words. At this point the Hypothesis Base has grown to 180 hypotheses, and the Case Base contains 1591 cases (sequences of 3 hypotheses). Some measure of English regularity might be derived from the fact that the 750 words PRO trained on could have produced roughly 4200 distinct cases if no repetitions of cases were present. (180 hypotheses are capable of generating 5,832,000 cases if all combinatorically possible combinations are considered.)

When we talk about PRO's performance stabilizing at a saturation point in training, we are assuming that the Hypothesis Base and the Case Base will eventually level out and cease to expand at any significant rate. To see how close we are to saturation after 750 words, we can examine growth curves for the Hypothesis Base (see table 2) and the Case Base (see table 3). Neither curve appears to be very level after 750 words, but it is clear that the Hypothesis Base is slowing down much faster than the Case Base. After their initial growth spurts, the Hypothesis Base acquires roughly 1 new hypothesis for every 10 training items, while the Case Base picks up 21 new cases for every 10 training items.

While it seems safe to assume that performance won't stabilize until the Hypothesis Base saturates, it is less clear that the Case Base must saturate before performance stabilizes. General correlations between memory and performance will have to be borne out by additional experiments, but if we must make some estimates, we could extrapolate from our existing growth curves and make some guesses about the state of PRO's memory when PRO's performance curves level off. Using the estimates for stabilization discussed above, it appears that PRO should acquire no more than roughly 240 hypotheses altogether, and performance will stabilize around 2800 cases. Note that we are assuming saturation of the Hypothesis Base by the time performance stabilizes, but we are less inclined to assume that the Case Base cannot continue to grow after performance stabilizes. We will only know this with certainty by expanding PRO's training corpus.

As for the estimates concerning the training vocabulary, we would guess that PRO will stabilize after about 1500 words (where a word contains an average of 3.64 phonemes - the same ratio

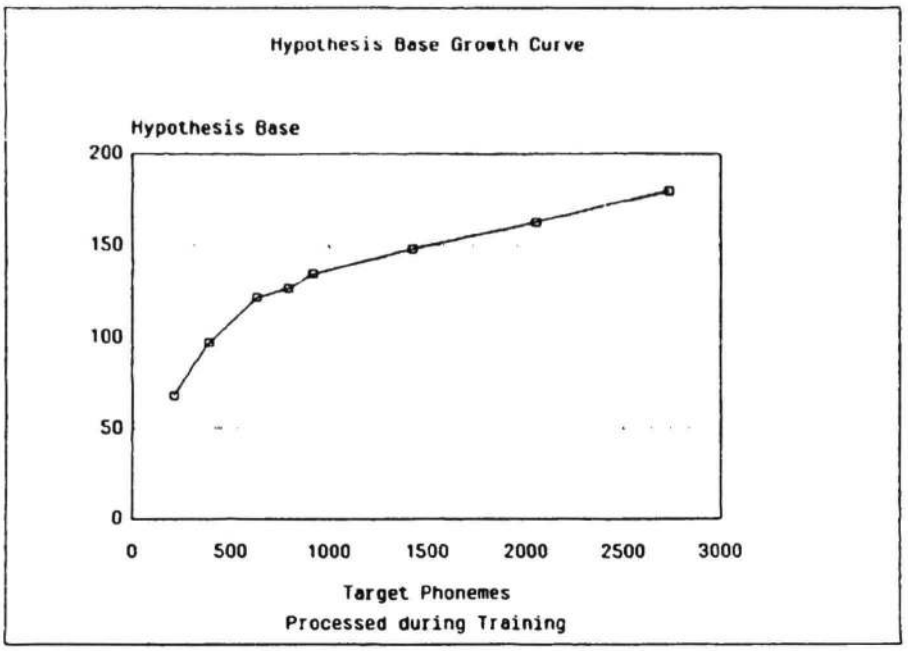

Table 2: The Acquisition of Hypotheses during Training

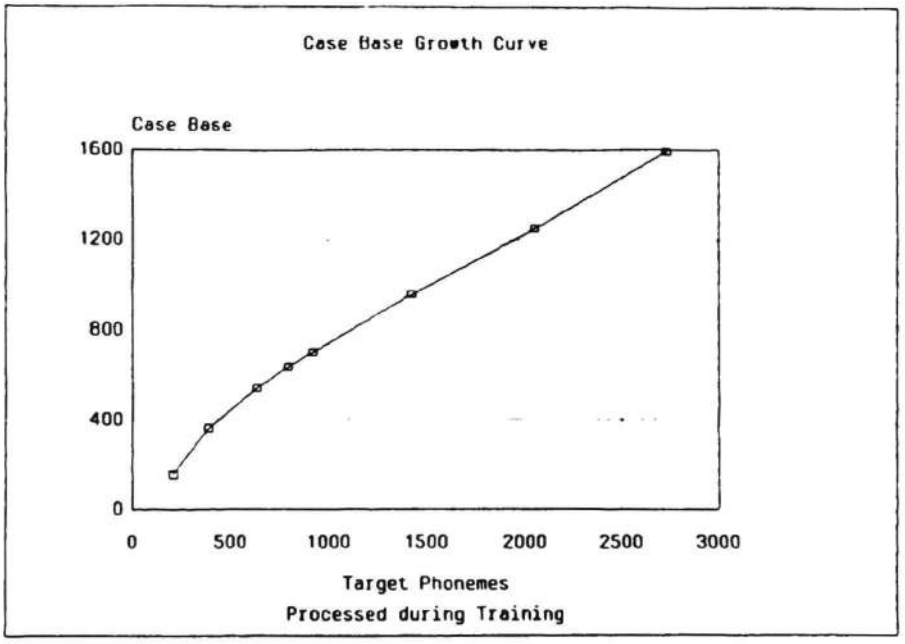

Table 3: The Acquisition of Cases during Training

maintained by our initial 750 word corpus). Of course all of these estimates are subject to the usual assumptions about random vocabulary and "representative" training sets. In selecting words from the dictionary we have tried to restrict ourselves to common words that any high school student would recognize. We cannot say how these restrictions relate to characterizations of English vocabulary in general.[2]

Looking inside the the Hypothesis Base, we find 47 phonemes and 101 graphemes underlying our 180 hypotheses. 69 hypotheses map from a single letter, 100 hypotheses map two-letter graphemes, 9 hypotheses map three-letter graphemes, and 2 hypotheses map four-letter graphemes. The single greatest source of ambiguity comes from the letter "a" which is associated with 10 distinct pronunciations (where "a" stands alone as a complete grapheme by itself). About two-thirds of the hypotheses have graphemes containing at least one vowel.

Conclusions

PRO shows how a large Case Base can be automatically acquired during supervised training and accessed within the framework of a spreading activation algorithm. PRO builds a three-tiered memory of hypotheses, cases, and frequency data using heuristic methods for knowledge acquisition. It is interesting to note that PRO is consistent with the concept of "timid acquisition" which is thought to be a necessary condition for success when training with a corpus of positive examples only [Berwick 1986].

A test item is processed by generating a search space based on available hypotheses and augmenting that search space with additional nodes derived from the Case Base. By structuring a network which represents both the search space and the influences of available memory, we can see in a declarative manner exactly how PRO views each test item in the context of its previous training experience.

Spreading activation and network relaxation then operate to consolidate competing information into a global consensus within the search space. While the structure of the search space is purely heuristic, the use of a relaxation algorithm to resolve the search is inspired by connectionist efforts. In this manner PRO represents a synthesis of ideas from both heuristic or symbolic methodologies and subsymbolic processing strategies.

References

Berwick, R.C. (1986) "Learning from positive-only examples: the subset principle and three case studies." in Michalski, Carbonell, and Mitchell (ed.), Machine Learning vol. 2. Morgan Kaufmann. pp. 625-645.

Bradley, L., and Bryant, P.B. (1983) "Categorizing sounds and learning to read: a causal connection." Nature, 301, pp.419-21.

Coltheart, M. (1978) "Lexical access in simple reading tasks." in Underwood, G. (ed.), Strategies of Information Processing. Academic Press: London.

[2] Although a claim has been made that the 1000 most common English words are also amongst the most irregular (see Sejnowski and Rosenberg 1986, p. 8).

Doctor, E., and Coltheart, M. (1980) "Phonological recoding in children's reading for meaning." Memory and Cognition, 80, pp. 195-209.

Feldman, J.A., and Ballard, D.H. (1982) "Connectionist models and their properties." Cognitive Science Vol. 6, no. 3. pp. 205-254.

Gough, P. (1972) "One second of reading." in Kavanaugh, J.P., and Mattingly, I.G. (eds.), Language by Eye and by Ear. MIT Press: Cambridge, MA.

Sejnowski, T. and Rosenberg, C. (1986) "NETtalk: A Parallel Network that Learns to Read Aloud." The Johns Hopkins University Electrical Engineering and Computer Science Technical Report JHU/EECS-86/01. Johns Hopkins University. Baltimore, MD.

Stanfill, C., and Waltz, D. (1986) "Toward memory-based reasoning." Communications of the ACM, vol. 29, no.12. pp.1213-28.